

To make sure that people have completed everything, you'll need to have a javascript file, in the <head> area of your page. To do this, first, let the browser know that what's coming is java script. So, after the background color, font info, etc, but before you leave the <head> section, put: <SCRIPT LANGUAGE="JavaScript">

If you want to only remind people to complete questions once, and then them leave questions blank, you need to declare variables, which will be counts of how many times people have gone through the page, being reminded. You'll want to create one of these counting variables for each question you have. An easy way to do this and keep everything straight is to use each variable name for each question, and tack on a letter at the end to indicate it's a counting variable. So, for instance, with the variable gender, the counting variable would be genderc.

Next, you want to declare each of these variables, and set them each equal to zero (since the first time someone comes to the page, each counting variable should be a zero. To do this, immediately after you've specified that what follows is JavaScript, you'll have one line for each counting variable, saying:

```
var genderc = 0;  
var var2c = 0;  
etc
```

Next, you want to create a function that will be run when the submit button is clicked. Since this function is validating the form, you can call it validate. First, name the function by saying:  
function validate()

next, define the function. Use a { to indicate that the definition is starting.

For each variable, you'll have an if statement. Essentially, each of these if statements will check to see if the answer is blank, and if the count of that variable is zero. If both of these are true, the focus of the page will go to the question that's blank, and an alert box will open, asking the participant to answer the question, and the count associated with that variable will be increased by one.

The check for a question being blank will differ depending on whether it's a text box answer, a radio button, or a pull down menu.

For a text box, checking whether it is blank will entail:

```
(document.forms[0].var name.value) == ""
```

For a radio button, checking whether it is blank will entail making sure that none of the possible values for that variable are checked. For some reason, no matter what you call the first value of a variable, in the eyes of JavaScript, the first possible value is zero. So, if you have a 1-5 scale of the variable nervous, checking whether one of those answers was selected would entail:

```
(document.forms[0].nervous[0].checked) == false &&  
(document.forms[0].nervous[1].checked) == false &&  
(document.forms[0].nervous[2].checked) == false &&  
(document.forms[0].nervous[3].checked) == false &&
```

```
(document.forms[0].nervous[4].checked) == false)
```

For a pull-down menu, checking whether it is blank will entail checking to see whether it is still equal to the default value (usually the first option, which JavaScript sees as having an index value of 0)

```
(document.forms[0].var name.selectedIndex) == 0
```

Putting the checking for no answer, with the checking for the first time through the page will look like this for a sample textbox, looking at a variable called hours:

```
if (hoursc == 0 &&  
    (document.forms[0].hours.value) == "")
```

For a radio button, looking at a variable called nervous it will look like this:

```
if (nervc == 0 &&  
    (document.forms[0].nervous[0].checked) == false &&  
    (document.forms[0].nervous[1].checked) == false &&  
    (document.forms[0].nervous[2].checked) == false &&  
    (document.forms[0].nervous[3].checked) == false &&  
    (document.forms[0].nervous[4].checked) == false)
```

For a pull-down menu, looking at a variable called gender, it will look like this:

```
if (genderc == 0 &&  
    (document.forms[0].gender.selectedIndex) == 0)
```

where 0 represents the first option in the pull-down menu (for instance, an option of “please choose one”).

Next, you need to tell the computer what to do if both of those things are true (it’s the first time through the script, and the question is unanswered). After the conditions of the if, put a { to tell it here’s what to do. You want to give an alert box, and put the focus of the form on the unanswered question, and increment the count variable (so it won’t remind people the next time through to answer the same question – to remind them every time, get rid of the count variable in the if statement). To do this:

```
{  
    alert("Please enter the number of hours you spent with your spouse.")  
    document.forms[0].hours.focus()  
    hoursc ++;  
    return false  
}
```

To put all this together, you don’t want to overwhelm the participant with alert boxes and refocusing of the questionnaire, so you want to step through one variable at a time. So:

```
function validate()  
{  
if (count variable 1 == 0 && variable 1 is blank) {  
    alert("Please answer variable 1")  
    document.forms[0].variable 1.focus()  
    variable 1c ++;
```

```

        return false
    }
    if (count variable 2 == 0 && variable 2 is blank) {
        alert("Please answer variable 2")
        document.forms[0].variable 2.focus()
        variable 2c ++;
        return false
    }

```

and so on, for each variable.

Finally, if none of those things are true (i.e., if no variables have a count of 1 and are blank), you need to tell the computer what to do. So, if none of your ifs are true, you need to give it an else to do. To do this:

```

else {
    return true
}

```

So far, you have a script that will give people an alert and return “false” if everything isn’t filled out the first time through, and give a “true” if everything is filled out, or if you’ve already reminded people.

Finally(!), you need to tell the browser you’re finished defining functions, and finished with JavaScript, by saying `</script>` and get out of the head and into the body of the page.

Ultra finally, when you have the submit button, you want the browser to carry out that validate script that you just created. To do this, rather than the usual for the submit button, you’ll have: `<center><INPUT type="submit" value="Submit" onClick="return validate()">`

So, if the validate function returns true, the responses will be processed; if the validate function returns false, those alerts will come up, and the responses won’t be processed until the participant either completes the questions or goes through all the alerts.

Unlike php scripts, you can test out JavaScripts without being connected to the internet, and without uploading the file (like HTML).